CLAIMS:

Having thus described our invention, what we claim as new, and desire to secure by Letters Patent is:

1. A method for performing a background code update of a current code image with an incoming code image in an embedded system, the method comprising the steps of:

(a) executing the current code image in the embedded system;

(b) executing one or more code update routines from the incoming code image to update the current code image with the incoming code image; and

(c) executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

2. The method according to Claim 1, wherein the method further comprises a step of retrieving an offset from the incoming code image for the one or more code update routines in the incoming code image.

3. The method according to Claim 1, wherein the method further comprises a step of retrieving an offset from the current code image of a task switching function.

4. The method according to Claim 1, wherein the method further comprises a step of loading all or part of the incoming code image into random access memory for execution.

5. The method according to Claim 1, wherein the method further comprises receiving the incoming code image into the embedded system via an input/output interface.

6. The method according to Claim 1, wherein the method further comprises the steps of:

providing a plurality of programmable memory devices for storing copies of the current code image;

executing a copy of the current code image from one programmable memory device; and

updating a copy of the current code image in other programmable memory device with the incoming code image.

7. The method according to Claim 3, wherein the method further comprises a step of testing the offset of the task switching function for validity before executing the task switching function.

8. The method according to Claim 1, wherein the method further comprises the steps of:

yielding microprocessor control by the executing function upon a task switching event; and

switching microprocessor control to continue executing the one or more code update routines to update the current code image with the incoming code image.

9. The method according to claim 8, wherein the method further comprises a step of continuing to switch microprocessor control between the one or more code update routines of the incoming code image and one or more functions of the current code image until the background code update completes.

10. The method according to Claim 1, wherein the task switching event is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

11. The method according to Claim 8, wherein the task switching event is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

12. The method according to Claim 1, wherein the method further comprises a step of resetting the embedded system upon completion of the background code update.

13. An embedded system for performing a background code update of a current code image with an incoming code image, the system comprising:

a first programmable memory device for storing the current code image;

a microprocessor for executing the current code image in the embedded system and for executing one or more code update routines to update the current code image with the incoming code image; and

a task switching means for executing a task switching function in the current image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

14. The embedded system according to Claim 13, wherein the embedded system further comprises a random access memory for loading all or part of the incoming code image for execution by the microprocessor.

15. The embedded system according to Claim 14, wherein the embedded system further comprises an input/output interface for receiving the incoming code image into the embedded system.

16. The embedded system according to Claim 13, wherein the embedded system further comprises a second programmable memory device for storing a copy the current code image, wherein the microprocessor executes the current code image from the first programmable memory device and updates the copy of the current code image in the second programmable memory device with the incoming code image.

17. The embedded system according to Claim 13, wherein the incoming code image instructs the microprocessor to further test the offset of the task switching function for validity before executing the task switching function.

18. The embedded system according to Claim 13, wherein the task switching means further switches microprocessor control from the executing function to continue executing the one or more code update routines to update the current code image with the incoming code image.

19. The embedded system according to claim 18, wherein the switching function instructs the microprocessor to switch control between the one or more code update routines of the incoming code image and one or more functions of the current code image until the background code update completes.

20. The embedded system according to Claim 13, wherein the task switching means is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

21. The embedded system according to Claim 18, wherein the task switching means is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

22. The embedded system according to Claim 13, wherein the embedded system further comprises a bootloader for instructing the microprocessor to execute the current code image and the one or more code update routines of the incoming code image, and to reset the embedded system upon completion of the background code update.

23. The embedded system according to Claim 15, wherein the embedded system comprises a bus for interconnecting one or more system components including

the microprocessor, the random access memory, the first programmable memory device, and the input/output interface.

24. The embedded system according to Claim 23, wherein one or more of the system components form a part of an integrated microprocessor.

25. The embedded system according to Claim 22, wherein the programmable memory device comprises a boot sector for storing the bootloader.

26. The embedded system according to Claim 22, wherein the bootloader tests the integrity of the current code image before instructing the microprocessor to execute it.

27. The embedded system according to Claim 22, wherein the bootloader is enabled to check for availability of a code update and if the code update is available to initiate the code update.

28. The embedded system according to Claim 13, wherein the current code image and the incoming code image include offsets within each respective image for code update routines and a task switching function.

29. The embedded system according to Claim 28, wherein the each respective code image comprises a header area for storing the offsets for the code update routines and the task switching function.

30. The embedded system according to Claim 13, wherein the offsets for the code update routines and the task switching function are stored at predetermined locations within each respective code image.

31. A storage automation library comprising an embedded system, the embedded system comprising:

a first programmable memory device for storing the current code image;

a microprocessor for executing the current code image in the embedded system and for executing one or more code update routines to update the current code image with the incoming code image; and

a task switching means for executing a task switching function in the current image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

32. A program storage device, tangibly embodying a program of instructions executable by a machine to perform a method for performing a background code update of a current code image with an incoming code image in an embedded system, the method comprising the steps of:

(a) executing the current code image in the embedded system;

(b) executing one or more code update routines from the incoming code image to update the current code image with the incoming code image; and

(c) executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

33. The program storage device according to Claim 32, wherein the method further comprises a step of retrieving an offset from the incoming code image for the one or more code update routines in the incoming code image.

34. The program storage device according to Claim 32, wherein the method further comprises a step of retrieving an offset from the current code image of a task switching function.

35. The program storage device according to Claim 32, wherein the method further comprises a step of loading all or part of the incoming code image into random access memory for execution.

36. The program storage device according to Claim 32, wherein the method further comprises receiving the incoming code image into the embedded system via an input/output interface.

37. The program storage device according to Claim 32, wherein the method further comprises the steps of:

providing a plurality of programmable memory devices for storing copies of the current code image;

executing a copy of the current code image from one programmable memory device; and

updating a copy of the current code image in the other programmable memory device with the incoming code image.

38. The program storage device according to Claim 34, wherein the method further comprises a step of testing the offset of the task switching function for validity before executing the task switching function.

39. The program storage device according to Claim 32, wherein the method further comprises the steps of:

yielding microprocessor control by the executing function upon a task switching event; and

switching microprocessor control to continue executing the one or more code update routines to update the current code image with the incoming code image.

40. The program storage device according to claim 39, wherein the method further comprises a step of switching microprocessor control between the one or more code update routines of the incoming code image and one or more functions of the current code image until the background code update completes.

41. The program storage device according to Claim 34, wherein the task switching event is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

42. The method according to Claim 39, wherein the task switching event is one selected from the group consisting of: round robin task switching; event driven task switching; and time slice task switching.

43. The program storage device according to Claim 32, wherein the method further comprises a step of resetting the embedded system upon completion of the background code update.

44. A method for performing a background code update of a current code image with an incoming code image in an embedded system, the method comprising the steps of:

(a) executing the current code image in the embedded system;

(b) retrieving an offset from the incoming code image of one or more code update routines in the incoming code image;

(c) executing the one or more code update routines to update the current code image with the incoming code image;

(d) retrieving an offset from the current code image of a task switching function upon a task switching event; and

(e) executing the task switching function to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.